Acts of digital parasitism: Hacking, humanitarian apps, platformisation

Introduction

The 'black box' has become a shorthand for how opaque digital technologies, platforms and devices have come to shape our social and political lives. Black boxes pose particular difficulties for the public understanding of the effects of complex digital mediation and democratic accountability. Frank Pasquale (2015) highlights the entangled emergence of black boxing through the proprietary nature of algorithms, software and digital technologies and coins the moniker of 'black box society'. Furthermore, black boxing needs to be understood as an effect of the production of digital technologies, which are not just proprietary. In the case of machine learning, it is the opacity of these technologies, at times to experts themselves, which also produces black boxing effects (Burrell 2016). Opacity emerges in human-nonhuman assemblages, as algorithms become black boxes through translations of calculative practices and 'decisions become encoded and encapsulated in complex inscrutable algorithms […], which after many iterations of ''bug fixing'' and ''tweaking'' even the programmers often no longer understand' (Introna 2015, 9).

The opacity of black boxes has also been a key challenge for researchers and developments in critical digital methods. On the one hand, much of the critical work on digital methods has focused on online sources that are openly available (Rogers 2013), even as the collection of data requires painstaking work of scraping and crawling. Digital methods have been defined 'as the repurposing of the inscriptions generated by digital media for the study of collective phenomena' (Venturini et al. 2018), thus relying on data that is visible, accessible or in some way made public. Increasingly opaque and complex digital ecosystems have received less attention. Moreover, opening black boxes can still prevent critical analysis there is no guarantee that 'readability' will be achieved. (Rieder and Röhle 2012, 113).

This article proposes a novel strategy to address the challenges of black boxes for critical digital methods by developing 'hacking' as a critical methodological device for digital humanities and social science research. While hacking has attracted a lot of attention in political engagements with digital media and digital technologies, hacking as digital method poses its own difficulties, particularly as hacking has been associated with 'high-tech guilds' and coding competences (Coleman 2001). Even as hacking could also be understood as a device for 'inventing the social' not limited to a class of technical virtuosos (Kelty 2018), it remains fraught with issues of opacity and exclusion of those who do not know how to code. Drawing on Michel Serres's work on the parasite (1982, 2017), we propose to reconceptualise hacking as 'acts of digital parasitism'. We thus take earlier suggestions to conceptualise the digital as parasitic (Berry 2016, Pasquinelli 2008) in a methodological direction. We argue that parasitic interferences allow us to address the methodological limitations of existing

approaches to opening the 'black box'. By recasting hacking as acts of digital parasitism, we develop hacking as a collaborative and interdisciplinary digital method that tackles the co-constitution of visibility and invisibility, transparency and opacity, access and restriction in complex digital ecosystems.

In order to unpack this methodological proposal, we focus on the accelerated production of apps for refugees by humanitarian actors. Apps are a particularly interesting object for digital methods research, as they are located at the uneasy intersection between transparency and opacity, being both 'open' interfaces and 'opaque' algorithms used on proprietary platforms. Humanitarian apps developed for refugees intensify these tensions, particularly as humanitarian actors need to understand what apps can do and as the apps target some of the most vulnerable groups worldwide. Critical scholars have already highlighted the political effects of digital technologies and big data used by humanitarian actors and the risks they can pose to the vulnerable populations whose protection they seek (for example Jacobsen 2015, Read et al. 2016, Burns 2015). The production of apps for refugees marked a new development in the practices of digital humanitarianism, which had started by harnessing the power of digital technologies and social media to respond to worldwide emergencies and disasters (OCHA 2017). The development of apps for the purposes of humanitarian governance presented us with renewed methodological challenges and political questions, as it led us to engage how digital methods problematise 'black boxes' and how they can come to 'compose problems as interruptions of the (historical) present' (Lury 2018, 3).

Our reconceptualisation of hacking as acts of digital parasitism emerged organically, through a series of collective methodological experiments, failures and surprises in the attempt to research (in)visible and complex digital ecosystems of humanitarianism. The paper traces this messiness of methodological work, while also configuring the more 'formalized enactments' (Law 2004, 122) that would advance a digital methods agenda. We start by locating hacking within methodological approaches to 'opening the black box' and explore its limitations through our first collaborative attempt to deploy hacking as a methodological device. While extending hacking to non-coders, this attempt led to an impasse, as hacking was often deployed to make visible security glitches and failures. In a second section, we trace the conceptual and methodological experimentation that led us to rework hacking as 'acts of digital parasitism' and, in a third section, discuss its implications as a critical method for researching the platformisation of humanitarianism and other digital relations. We conclude with a set of suggestions for further work using hacking as methodological interference for digital humanities and social science research.

Opening black boxes: controversy mapping, reverse engineering, hacking

The metaphor of the 'black box' long precedes our digital 'black box society'. Borrowed from cybernetics, the metaphor came to inform methodological developments in Science and Technology Studies (STS). 'Opening the black box' has been widely invoked in STS research

and is famously associated with the work of Bruno Latour and Michel Callon. For Callon and Latour, black boxes are metaphors of power, as they contain 'that which no longer needs to be considered, those things whose contents have become a matter of indifference' (Callon and Latour 1981, 285). Thus, opening black boxes entails a project of understanding how the technology works, with which assumptions and effects. In his earlier work, Latour proposed to open the black box by 'moving in time and space until one finds the controversial topic on which scientists and engineers are busy at work' (Latour 1988, 4). Opening the black box is about finding the sites of controversy and collaboration. This was the 'back door of science in the making' rather than the existing black boxed and uncontroversial, taken-for-granted science. When these controversies are not accessible (any longer) to participant observation, other STS scholars suggest interviews as a device to open the black box. For instance, Donald MacKenzie elevates interviews as a central method to open the black box of finance, as he confesses that 'I cannot imagine discovering the history of nuclear missile guidance, or understanding the contingencies affecting the practice of arbitrage, without interviewing missile guidance engineers or arbitrageurs' (2005, 570).

Key to 'opening the black box' in STS is mapping controversies. While the study of controversies spans the history of STS and takes different methodological instantiations, digitisation has led to new innovations in controversy analysis (Venturini 2012). Controversy analysis or mapping deploys 'digital techniques for the capture, analysis, and visualization of—often Internet-based—data in order to render legible disputes about public issues' (Marres 2015, see also Marres 2017). Mostly deployed on social media platforms, controversy mapping relies on the very possibility of 'crawling and scraping' or on the availability of data.[1] Therefore, opening black boxes by detecting or tracing controversies does not fully address the opacity of diffuse digital ecosystems, particularly as machine learning, algorithms and big data make the connections between controversies and their stabilisation less traceable.

A second methodological device of directly tackling the black box has been captured through the specific coinage of 'reverse engineering'. Reverse engineering has been suggested as a methodological intervention to research what 'remains black-boxed' (Kitchin 2017, 19). Rather than a lateral intervention that traces controversies preceding the process of black boxing, reverse engineering is 'the process of articulating the specifications of a system through a rigorous examination drawing on domain knowledge, observation, and deduction to unearth a model of how that system works' (Diakopoulos 2014). While this can be useful, results often rely on the assumption that the system works as designed. Moreover, reverse engineering does not adequately recognise the 'networked and entangled nature of algorithms' (2016, 91). It thus becomes more difficult to account for 'the flow of information through a system and the choices and decisions that create, modify, and interpret the flow' (Hayles 2017, 116). Nick Seaver (2014) points out that reverse engineering can be unspecific in its results, as 'it misses the accidents that happened along the way — the abandoned paths, the unusual stories behind features that made it to release, moments of interpretation, arbitrary choice, and failure.' In its focus on reconstructing a finished system of code, reverse

engineering risks missing these failures and contingent choices and, by assuming that a system works, it aims to bring out the 'secrets' of what algorithms do.

If reverse engineering emerged particularly as a response to the limitations of transparency in relation to algorithmic accountability, it has also become the respectable foil of 'hacking', especially as hacking has been increasingly 'tarnished by fear' (Galloway and Thacker 2007, 77). Thus, hacking has been largely missing from the literature on critical digital methods, remaining the focus of sociological and ethnographic accounts of 'guilds' and coders. In this literature, hacking has been generally understood as the 'reordering the technologies and infrastructures that have become part of the fabric of everyday life' (Coleman 2011, 515, also Jordan 2017). Kate Crawford has proposed to reconnect reverse engineering and hacking by paying attention 'to the ways people reverse engineer algorithms, acting in direct contestation, where the troll and the hacker become key players in an agonistic system' (Crawford 2015, 13). While Crawford's suggestion relies on the hacker's specific technical competence, more recent developments in digital humanities have started to emphasise hacking as a methodological device or interference with digital technologies, which is extended to include non-coders and foster the agency of collective users (e.g. Pybus et al. 2015, Kelty 2018). Here, hacking the digital world is presented as targeting data 'leaks' through the development of methodological devices and collective research processes.

Building on these literatures, we propose to explore hacking as a socio-technical methodological device, which works through transversal connections between multiple actants and 'a seeing-from-within, a making-as-exploiting' as a mode of critique (Kelty 2018, 291). Similar to controversy analysis in STS, we see hacking as interdisciplinary, bringing together different social, political and computational competences. For us, hacking is also a critical method, as it can produce interferences in how digital worlds are continually (re)composed. Therefore, we also address the limitations of opening black boxes for critique in response to the concern that the rendering algorithmic black boxes transparent 'does not provide the means to position oneself critically towards what is inside the black box' (Paßmann and Boersma 2017, 142).

Our first experiment with hacking as a methodological device started with an interdisciplinary collective to explore data 'leaks': a digital humanist, a geographer, an International Relations scholar, a media studies scholar and a programmer, thus taking 'hacking' beyond its autonomous and high-tech spaces. This motley collective also included the DroidDestructionKit, developed to enable the 'hacking' of Android apps and to examine their practices and network traffic.[2] The DroidDestructionKit brings together several tools for decompiling and examining Android APK packages. The Android Package Kit (APK) is the package file format used by the Android operating system for distributing apps. The DroidDestructionKit is assembled into a virtual Linux machine and thus runs on all platforms. Its main aim is to repurpose classical security tools such as penetration testing into a research device that is useable not just by 'high-tech guilds'. The DroidDestructionKit integrates tools to unpack and analyse .apk files via an easy-to-use graphical interface. The virtual machine

includes, for instance, Wireshark, a widely-used network protocol analyser, or tcpdump, a common packet analyser, as well as several tools to examine the source-code of Android .apk packages. We packaged these tools into a VirtualBox in order to make it easier for non-coders to use these often command-line based tools. This also allows for easier installation of the required software, which can otherwise be a complex process, often requiring building it from source-code. The DroidDestructionKit can thus become a methodological device deployed in collaborations that allow researchers from multiple disciplinary backgrounds to experience 'hacking' Android apps, without being able to code.

We had collected a list of apps developed by humanitarian actors for refugees. Our interest was to understand the diffuse and disperse flows of data, the 'leaks' that are intrinsic to digital ecosystems. At first sight, the refugee apps developed by humanitarian actors appear as 'black boxes' because they are developed in cooperation with 'tech for good' commercial endeavours or simply because they are opaque to those who do not code and produce software. But they are also unique in that they are produced by organisations which are traditionally not linked to app development, but started developing apps as a new mode of connecting and communicating with refugees. Humanitarian app production for refugees has presented itself to us as a messy assemblage of entangled data, algorithms and devices rather than a single black-box or planned system. We wanted to attend to these opaque connections and complex data flows in relations between humans and non-humans. For us, hacking – understood as experimental, messy open-ended coding and making rather than breaking into computers and networks – could thus be a methodological device to understand the performativity of apps, what they do within disperse digital ecosystems rather than seen as informational content.

From a list of refugee apps, we randomly selected Refugee.info, developed by International Rescue Committee (IRC) for refugees on the so-called Balkan route in Europe.[3] Refugee.Info was then available on Play Store for Android and App store for iOS.[4] The app contained basic information services by the IRC for refugees and at the time gave users three options in terms of their location: Serbia, FYR of Macedonia and Greece. As with most refugee apps, the visible content of the app is directly informational: it lists emergency contacts, information about accommodation, legal aid, phones, wifi and communication tools. It also lists the transit and reception centres where free wifi is available. Yet, it is not this visible communication of the app with its target audience of refugees that we wanted to understand, but the data 'leakage' or opaque exchanges that were part of the humanitarian assemblage, while remaining unknown to refugees and perhaps to humanitarian organisations themselves.[5]

As lines of code appeared on the system and in it locations, the programmer – let's call them N – exclaimed in disbelief. 'This app is sending location in plain text' about where we were at https://www.google.co.uk/maps/@51.5084054,-0.1215376,16z.' It would be a dreadful oversight to transmit credit-card details this way, let alone the locations of the vulnerable. I can't estimate the likelihood of some "bad actor" lurking on the network and harvesting the information, but mitigating this would be trivial. Just add an "s" to the "http"

in the code (…)'. The enormity of the security lapse temporarily interrupted our collaborative exploration and vectored it towards 'corrective' security intervention. N insisted on contacting the app's developers first before we continued with the research and made the security lapse publicly visible: 'I added a comment on the Play Store, (…). There's been no response from @theIRC at all.' The technology could be corrected on the assumption that only adding an 's' to 'http' would have safeguarded the location information of refugees.

It was not this flow of location data that was most problematic, even as we were looking into data 'leakage'. Rather, it was its 'plain text' form and the lack of encryption by the https standard, which is easy to do but demonstrates a complete technical neglect of basic data protection standards by the app developers and the NGO. N's Play Store intervention led to the desired change a couple of weeks later and a new software release, which used the https encryption. Yet, this corrective intervention was less than straightforward, requiring interventions in public forums – from Twitter to posting a review of the app.

Our surprising discovery of leaked unencrypted data by Refugee.info oriented us away from the opaque but mundane data 'leakage', which had been our starting point for studying refugee apps. Furthermore, there was an easy technical fix, which had been oriented by the excesses of digital technology rather than a socio-material exploration of digitally mediated, mundane relations. To continue our methodological experiment, we needed to rethink mundane data leakage as intrinsic to digital ecosystems and digital humanitarian assemblages. We also needed to rethink our focus on hacking an app in order to understand the connectivity of apps within humanitarian practices. Our initial intervention did not dispute or eliminate data 'leakage'; it only 'corrected' its most egregious form. Our methodological device did not centre on practices themselves but only on their excesses. Therefore, we needed to move from representations of exception and transgression in digital technologies to mundane and messy practices of connecting. To do so, we proposed to reconceptualise both refugee apps and our methodological devices as 'parasitic'. The next section traces how this reconceptualisation of 'parasitic' technologies helped us develop a different perspective to the exceptional practices revealed by our first methodological intervention.

Parasitic technologies/Acts of parasitism

Our initial experiment with hacking as a methodological device led us to reconsider hacking as a messy practice that interferes rather than intrudes, and which works-alongside rather than corrects based on a diagnosis of vulnerabilities and excesses. Hacking then gives substance to the diffuse relationality that is implied in digital practices of assembling, networking and connecting. As the geographer John Allen (2011, 156) aptly pointed out, we need to specify the 'content of the relationships that hold assemblages in place'.

Michel Serres's (1982) concept of the parasite helped us develop a different understanding of relations in the digital-humanitarian assemblage and of our own methodological interference. Serres moves beyond the negative and moralising understanding of the parasite, often associated with its biological meaning of a microbe or

infection: '[t]the parasite is also a guest, who exchanges his talk, praise, and flattery for food. The parasite is noise as well, the static in a system or the interference in a channel' (1982, xi).

These meanings of the parasite led Serres to recast the understanding of relations (predator/parasite), exchange (use/abuse) and communication (signal/noise). What brings these three meanings together is the understanding of parasite as introducing a 'third' in an operation or a relation (Serres 1982, 198, 206). Serres developed his reading of the parasite drawing on a mixture of sources, from La Fontaine's fable *The Town Rat and the Country Rat* and ancient Greek comedy, to biology and information theory. In Greek comedy, the parasite is the character allowed to eat at the host's table by entertaining the guests 'with his stories and his mirth' (Serres 1982, 34). Etymologically, the parasite means to 'eat next to'. For us, the meaning of 'next to', of being alongside rather than against is important for reconceptualising hacking as a critical digital method. Serres distinguishes the parasite from understandings of exchange to show that the parasite enacts a different mode of relationality:

> The parasite invents something new. Since he does not eat like everyone else, he builds a new logic. He crosses the exchange, makes it into a diagonal. He does not barter; he exchanges money. He wants to give his voice for matter, (hot) air for solid, superstructure for infrastructure (Serres 1982, 35).

For us, parasitic relations are transversal in the sense in which they introduce a 'third', a surplus which modifies existing relations by working-alongside or beside rather than working-against them. The ambivalent meaning of the parasite makes possible a conceptualisation of digital technologies as parasitic rather than predatorial, as asymmetric reciprocity rather than equal exchange and as noise rather than just communication. Communication is thus itself never binary and refugee apps do not simply connect humanitarian organisations and refugees. Data leakage is not the inadvertent by-product of a binary model of economic exchange, but it is the parasitic constitution of ternary relations. Based on this understanding, we need to attend not to the data that is contractually exchanged in the process of setting up the use of digital technology, but the data that appears as 'noise', a surplus to digital relations which simultaneously makes these relations possible. Noise is not an interruption, but an interference that 'transforms the nature of the message depending on the position of the receiver and their activities' (Brown 2013).

Approaching digital relations as parasitic allows us to reconfigure the understanding of hacking as predatorial, agonistic or 'against', as a practice that corrects technical shortcomings. As a methodological device, hacking can be understood as becoming 'noise' within any digital communication and in particular apps in order to discover relations these apps are part of. It is also interdisciplinary, assembling diverse competences and thus a transversal socio-technical interference. Working-alongside and assembling are not devoid of friction, but they do not work on the binary model of exchange, antagonism or agonism. Inspired by Serres's concept of the parasite, we propose to conceptualise these

methodological interferences as 'acts of digital parasitism', as they disrupt in subtle ways, through small interferences akin to 'noise' rather than through spectacular intervention.[6]

A few months after the first methodological experiment, we arrived at this understanding of parasitic methodological interference as a result of unexpected events, the expansion of our interdisciplinary collaboration and – most importantly – as a direct result of opening hacking to users outside of 'high-tech guilds'. Collaborating with humanitarian organisations during a workshop, we came to identify how 'hacking' could become a device to add and expose noise in the digital-humanitarian assemblage. This interference expresses a different mode of relationality rather than that of improving technical communication or addressing vulnerabilities by 'taming' technological excesses or flaws.

As part of this workshop, our collective was composed of humanitarian professionals, students, academic researchers, media experts and most of our original team. The workshop was organised around position papers on the current situation of refugees and their use of digital technologies, theoretical examinations as well as practical experiments using the DroidDestructionKit. Everybody had been asked to download the toolkit on their laptops in advance and had been provided with our list of refugee apps. A critical part of the toolkit is the open-source Raccoon software, an APK download software, which made the data gathering possible.[7] A couple of days before the event, Raccoon began to fail due to 'a recent change by Google to how apps have to log in to Play' (Ahlbrecht 2017). In the words of Raccoon's developer, 'This isn't the first time Google breaks Raccoon. It won't be the last time either' (Ahlbrecht 2017). With Raccoon, it would have been relatively easy to download an Android application onto any device and to extend hacking to coders and non-coders. In its absence, we were limited to using the Android applications directly on the mobile phone that had to also be 'rooted', allowing applications to be run with administrative privileges.

On the day of the workshop, we managed to work around Raccoon by using the Google Play Store of a mobile connected to the computers to download the apps directly. It was this breakdown of the 'hacking' workflow with the DroidDestructionKit that led us to approach 'hacking' as becoming noise in the digital infrastructure, working-alongside and within existing digital relations and data exchanges. On a better day for the hacking parasite, Raccoon would deliver the APK package to any device. APK files are similar to compressed ZIP files, and therefore are easy to extract and examine using the tools that the DroidDestructionKit provides. Once unpacked, a typical research workflow targeting data leakage starts with Android Manifest as a first point of contact to provide orientation. AndroidManifest.xml is a human-readable declaration file that must be provided by an app's developer and contains essential information about the app as required by Google Play Store.[8] Every interactive user-interface (activity) and background process (service) are listed, as well as so-called 'BroadcastReceivers' that receive inter- and intra-app messages.

As this workflow was interrupted by Google's breaking of Raccoon, we could only move forward by shifting from the app to the operating system. This required the use of 'rooted' devices and limited the overall time we had at the workshop. These accidental interferences also moved us away from data leakage towards the different relations the app

is embedded in. Given the temporary failure of Raccoon, the mobile phone environment we were moved into did not allow for extensive searching through the code for location definitions or other data 'leaks'. As only one of the participants at the workshop managed to decode the app on the day of the workshop, we then started working collectively on the same code that was projected onto the screen.

The programmer took the lead going through the code and quickly skipped what appeared to be familiar lines of code. These included the Application Programming Interface (API) calls to outside web and software services from the big Silicon Valley names, which enable standard and advanced app behaviour like map visualisations but also dedicated services such as the delivery of opening times of refugee centres. This is not unusual for any app, which is why the programmer skipped those lines, but it was unexpected for others, and especially the NGO participants. While they were aware that Google provides map services for apps, the scale of integration and thus dependency upon the big Internet companies beyond Google on a deeper techno-relational level surprised them. We stopped the programmer at the relevant code and no longer continued the search for data 'leaks'. We shifted from what experienced programming eyes consider normal in code to a different collaborative inquiry into what counts as mundane practice. The DroidDestructionKit supports these interferences by including tools to 'de-compile' and analyse apps.

Most Android applications are written in Java. Their code, however, is not compiled to the typical Java .jar files for execution by the Java Virtual Machine, but to .dex files, which are executed by the 'Dalvik' virtual machine, now the 'Android Runtime' (ART) on Android versions from 5.0 onwards. Our DroidDestructionKit provides a utility called 'dex2jar' that converts the .dex files from the APK package into .jar files, which in turn can be decompiled into Java code with a program called JD-GUI.[9] Valuable insights about the app can be gained without any ability to read Java code, for instance from Java class names or methods. While descriptive names for variables are often lost in the necessarily imperfect de-compilation process, in our experience names of Java classes and their method signatures are often readable, as are character strings such as URLs for third-party APIs.

In the first methodological experiment, the programmer searched for character combinations in the code that are known to indicate invasive sources in source code. A small subset of Java classes and APIs is associated with potentially invasive sources of data such as location, SMS messages, call logs and wireless network connections. Searching for these in the source code can determine which third-party libraries access which features. In the second collaborative workshop, our opening to the NGO humanitarian community and to researchers from different disciplines led us to move away from targeted searching of 'leaks' towards an open-ended browsing of the source code. This new collective focus on app practices shifted attention to the 'noise' of external libraries from Google, Facebook, and Airbnb embedded in the refugee apps and mediated by Application Programming Interfaces (APIs).

APIs are the critical infrastructure for app production, as a digital ecosystem is only possible due to APIs (Blanke 2014, 44). Originally, an API helped a single computer connect to

its peripherals such as printers. Nowadays, APIs are web-scale and connect anything to anything, while simultaneously locking in developers and users to the world of the API provider. The provision of APIs thus leads to the platformisation of the web in both an infrastructural and economic sense (Helmond 2015). Google is still one of the best examples of how APIs are the prime drivers of platforms, as they both enable and orient digital relations. Even Apple did not manage to permanently remove Google Maps from its own ecosystem (Blanke 2014, 44). In iOS 6, Apple tried to replace Google Maps but then had to apologise to its users for subsequent bad service. Since then, Google Maps have maintained market dominance with API lock-in. Through platformisation, APIs also become the parasitic mediators of digital relations of asymmetric reciprocity. Thus, following Serres, APIs cannot be eliminated but can only be incorporated or parasite themselves. The final section explores the implications of API parasitic relationality for hacking.

Parasiting the parasites: platformisation and humanitarianism

Parasite logic never stops. (Serres 1995, 58)

The advantage of formulating hacking as acts of digital parasitism is that parasites are neither predators nor transgressive actants. The aim of a parasite is not to destroy a system or correct (the excesses of) a technology. Acts of digital parasitism as methodological interferences are about operating on relations rather than corrective actions, as we did when we moved from targeting invasive sources to browsing through API relations. This new understanding opened up parasitic relations in the app-based humanitarian ecosystem. It allowed us to focus on the socio-materiality of apps by bringing together the analysis of parasitic digital relations with the digital materiality of code.

Our understanding of hacking as critical method was also transformed through our collective assembling. In the collaboration, we experienced the outside connections to APIs that 'lock in' the app's companies or, in our case, NGOs and their users into a long-term co-existence with the provider of the API services and data. NGOs will commit to an API without even realising it, as its software development is often outsourced. Humanitarian NGOs are typical for a new generation of organisations, which have not been involved in software development, but now need to build digital connections. At the same time, they add their own specific configurations to software practices. In the case of humanitarianism, apps are often the result of 'tech-for good' engagement activities rather than purely commercial developments. As a result of their digital transformation, the NGOs' future is mediated indefinitely through platforms whose architectures and code are developed for them. Over the past few years, an uneven infrastructure of services on the Web has developed with long-term consequences. This can also be experienced in the workings of humanitarian apps. In this section, we explore how our parasitic interferences can help trace the emergence of

‘platform humanitarianism’, a humanitarianism that does not use digital technologies, but which is enacted through parasitic relations on platforms.

In order to move from one app to relations between apps, a few months later we revisited the excitement of the NGOs during the workshop about the emergence of host-parasite and parasite-parasite relations. By 2018, Raccoon had been updated to catch up with the Google API it had lost contact with. However, the Google parasite experience had changed Raccoon, which meant that we had to adjust our toolkit to be able to automatically download apps. Between the search function of Google Play Store and an online directory of apps for refugees, we selected a sample of 18 apps developed by or in collaboration with humanitarian actors.[10] Our reconfigured toolkit decompiles the target apps in bulk and loads them to a graph database to capture all the relations of the sample apps. Rather than randomly selecting a specific app, we investigated API relations in a larger, but still focused, sample and traced the relations within the platforms their apps belong to.

The first result was a list of permissions for each app, which contain broad access to the networks and storages the mobile is exposed to, location information as well as billing and download information. We found several noteworthy entries. For example, C2DM stands for Cloud to device messaging. Apps need to send regular license checks to the server to verify that they are permitted to continue. About a third of the apps we researched demand locations, as the following table indicates:

COUNT	name
15	"android.permission.INTERNET"
14	"android.permission.ACCESS_NETWORK_STATE"
9	"android.permission.WRITE_EXTERNAL_STORAGE"
9	"android.permission.WAKE_LOCK"
8	"com.google.android.c2dm.permission.RECEIVE"
6	"android.permission.ACCESS_FINE_LOCATION"
6	"android.permission.ACCESS_COARSE_LOCATION"

Figure 1 represents the graph summary of the app ecosystem we have traced, where the connections stand for the frequencies with which APIs co-occur. We targeted both software and web APIs.

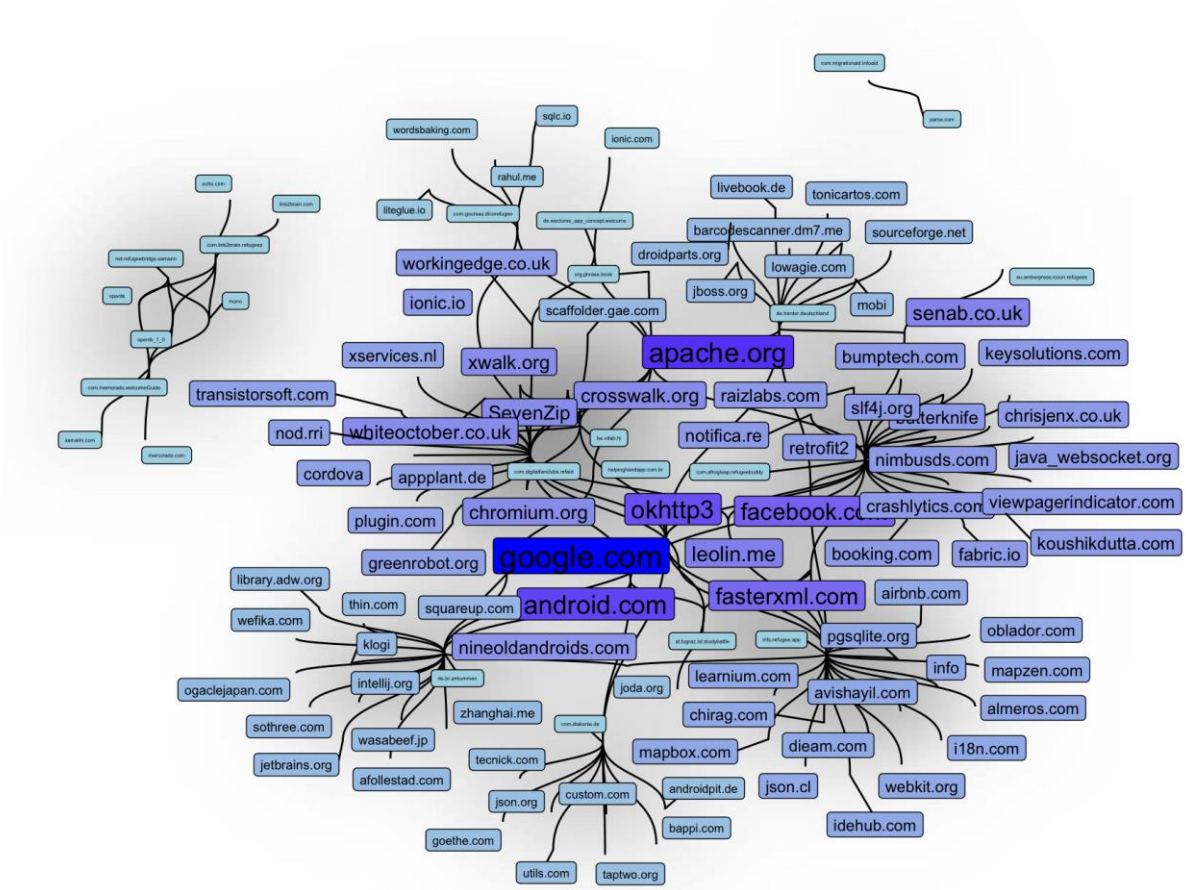

Figure 1 Network visualisation of APIs

The size and shade of the nodes in the network correspond to the number of APIs that are connected to a particular API in the apps' code. We can see that Google and Facebook dominate the relations that underpin refugee app production. Other software APIs that hold nodal positions are used in the day-to-day development of apps. Apache.org, for instance, relates to open source tools provided by Apache Software Foundation projects and commonly used in development. fasterxml and okhttp3 respond to specific common challenges in app development.

The connectivity of APIs can be read as a relation of mutual parasitism. Let us take Google, which is at the centre of the digital humanitarian assemblage we have researched. Google does not only 'host' searches or maps for the refugee apps, but it simultaneously intercepts data flows, it becomes a parasitic operator on digital relations. Similarly, the presence of Facebook in these humanitarian apps is far beyond what the representation of Facebook as a platform that one joins as a user. As the creator of Raccoon has recently quipped 'We are all Facebook users!' (Ahlbrecht 2018). Through the production of apps, digital humanitarianism becomes platformised. Understood in a broad sense as digital arrangements that connect users and organisations, platforms are also mediating and

enacting parasitic social and economic relations. As Plantin et al (2018) have noted, platforms need to be '"open enough" to generate a whole ecosystem of applications (e.g. Google maps mashups), but possess as end goal to simultaneously position themselves at the centre of such ecosystem, to eventually become the entity that regulates data circulation'. Facebook and Google are not only positioned at the centre of an emerging humanitarian ecosystems, but they reshape digital humanitarianism through parasitic relations, which we can research by inserting ourselves as another parasite into these relations. The companies of the centre of platforms provide part of the infrastructure of digital humanitarianism and are simultaneously 'an extractive apparatus for data' (Srnicek 2017, 36).

However, these nodal relations are not the only important ones. As platforms enact relations between different actants, we need to attend to the more isolated APIs as well. In our experience, these are signs of app development that was not conducted in-house. The digital humanitarian ecosystem is full of these. Wordsbaking.com, e.g., is used in Java code to record audio. It is now depreciated, as far as we understood the Chinese source site.[11] chrisjenx.co.uk stands for an individual Android developer. They are not the only developer to include themselves in the source code, which is a practice that was new to us. Booking.com also seems to provide services to humanitarian apps. It is not immediately clear why booking.com would be relevant for refugee apps, as it connects to the application's online listings on booking.com.[12] A possible explanation is provided by the reuse of the code in the software companies that developed the apps. Booking.com is a reminder that these apps are not developed in isolation but in conjunction with other app projects and platforms. As late comers to the world of app development, humanitarian actors are led to reuse these earlier technical building blocks in a form of digital bricolage.

One of the most interesting discoveries for us, however, were the emergent relations between parasites. As parasites cannot be eliminated, they can either be incorporated or replaced by other parasites. To put it in Serres's terms, only noise can supplant noise. For a long time, APIs have been used by developers to avoid reinventing the wheel by re-using external services such as Google's map provisions. Thus, APIs act as mediators not just for digital humanitarianism but for apps. In this logic, what matters for the parasite is to be placed 'in the most profitable positions, at the intersection of relations' (Serres 1982, 43). The vitality and scale of the digital ecosystem is indicated by the number of parasites involved. We are currently witnessing an API race by all actors in the digital ecosystem with ever increasing new layers of parasitic attachments. APIs become parasites of other APIs.

For instance, it surprised us to find Airbnb feature so prominently in the codes. We soon discovered that this is the case because it provides Google Map API services for countries and other locations where the majority of Android devices is sold without the core Google Play Services. While Google Services are sometimes not accessible, interactive maps remain a necessary feature of any Android experience. Airbnb has created an open source software – AirMapView – which can 'choose by default the best map provider available for the device' (Petzel 2015). AirMapView is adaptable to future additions such as Amazon Maps, Baidu or Mapbox. This new technological development parasites the Google parasite by disrupting its

original message and enabling new communication by choosing the best possible map provider available for a given location. It will use the mobile API where Google Play Services are available and otherwise the default web setting. Although platforms would appear to have a tendency towards 'enclosure as a key means of competing against their rivals' (Srnicek 2017, 83), we see platforms as increasingly characterised by parasitic relations where a parasite 'chases' another parasite.

Methodological interferences through 'acts of digital parasitism' enact our object of research as messy, surprising and unpredictable. When digital relations are understood as parasitic, APIs do not just help 'lock-in' users and solidify data extraction but aim to parasite other API parasites by situating themselves at the intersection of digital relations. At the same time, hacking refugee apps also shows a humanitarian ecosystem that is inhabited by depreciated APIs, APIs out of date or APIs that record the labour that would have otherwise become invisible.

Conclusion

Given the limitations of 'opening the black box' and methodological devices deployed to open the 'black box' of digital technologies, this article has developed interdisciplinary methodological experiments to analyse and understand what apps do. Apps have been particularly apt objects of research as they combine elements of transparency and opacity, visibility and invisibility and raise questions about data extraction and capture in platformised digital ecosystems. Their lack of transparency also poses challenges for digital methods research.

We have proposed 'hacking' as a methodological device to develop interferences with digital technologies through open-ended (re)coding in transversal collectives rather than transgressive intrusions by 'high-tech guilds'. Drawing on Michel Serres's work, we have reconceptualised 'hacking' as 'acts of parasitism', which create parasitic interferences by working-beside or alongside digital technologies and assembling collectives of coders and non-coders. Our reconceptualisation of hacking also mobilised STS understandings of digital technologies, code and apps as socio-technical relations. Hacking as 'acts of parasitism' was thus a subtly disruptive socio-technical interference in digital ecosystems, specifically humanitarian ones. As humanitarian actors and NGOs have increasingly come to produce apps to communicate and connect with refugees, we were interested in both the commonalities and singularities of their digital production and the effects of apps in enacting digitally mediated relations.

Initially starting with assumptions about data 'leakage' and extraction by digital technologies, we were led to focus on the exceptionality of unencrypted data leakage. While 'hacking' initially oriented us towards exceptionality and transgression, we subsequently realised that we needed to reconceptualise digital technologies in order to explore the habitual workings of refugee apps. Our experiments were characterised by failure, glitches and surprise, as much methodological work is. The reconceptualisation of digital technologies

as 'parasitic' and of hacking as 'acts of digital parasitism' led us to refocus our methodological interferences upon noise and becoming noise within digital relations. Rather than a black box that could be reverse engineered, the refugee apps we were researching could be better understood through parasitic relations, where the 'noise' of data extraction and APIs is reorienting communication and participation on platforms. While Google and Facebook APIs appeared central to the humanitarian ecosystem, these platforms were also messy and noisy, containing various depreciated and out-of-date APIs.

The methodological interferences developed here can be used to research apps more generally and advance digital methods for digital humanities and social science research. Acts of digital parasitism emerge through motley interdisciplinary collectives of coders and non-coders, which also include nonhuman actors – in our case, the DroidDestructionKit, which can be downloaded and experimented with in further research. However, unlike computational research that highlights the existence of trackers in over 90% of all apps on Google Store (Binns et al. 2018), research in digital humanities and social sciences is attentive to both specificity and contingency in socio-technical relations. Thus, our methodological interferences also bear particular significance for how we understand the platformization of humanitarianism and its app production. Humanitarianism is a particularly vulnerable site of data extraction, given its lack of in-house digital expertise, funding through 'tech for good' initiatives, and lack of capacities to parasite other parasites. Not only is humanitarianism more readily parasited given its sites of digital production, but it cannot tackle the depreciation, messiness and obsolescence of APIs and software. These contingencies of digital platforms and their effects for actors who do not command the resources of big Sillicon Valley companies need to be investigated further.

Notes

[1] On methods as devices see, for example, Law and Ruppert (2013), Aradau and Huysmans (2014).

[2] The DroidDestructionKit is distributed with a VirtualBox appliance and can be freely downloaded from https://github.com/kingsBSD/DroidDestructionKit.

[3] The Balkan route was used mainly by Syrian refugees arriving in Europe in 2018. It started in Turkey and continued through Greece or Bulgaria towards Hungary, Austria and then Germany.

[4] https://play.google.com/store/apps/details?id=info.refugee.app&hl=en. As of 2018, the Refugee.info app is part of the Signpost project, a collaboration between the International Rescue Committee, Peace Geeks and Mercy Corps, which also includes Italy. The app is now a web version only, as the mobile phone version has been discontinued.

[5] The 'walkthrough method' could be used to analyse the app's visible interface (Light et al. 2018)

[6] While similar to methodological devices in their creative enactments, methodological acts emphasise disruption rather than simply making a difference (Aradau and Huysmans 2014).

[7] http://raccoon.onyxbits.de/.

[8] https://developer.android.com/guide/topics/manifest/manifest-intro.html.

[9] https://github.com/java-decompiler/jd-gui.

[10] Apps for refugees lists more than 40 apps http://appsforrefugees.com/.

[11] https://github.com/vilic/wordsbaking-plus-chrome.

[12] https://news.booking.com/bookingcom-launches-new-content-api-for--accommodations-channel-managers-and-content-providers/.

References

Ahlbrecht, P (2017) *Let's talk about Raccoon v4.1.1* (10 June 2017) 2017 [cited 12 June 2017]. Available from https://blog.onyxbits.de/lets-talk-about-raccoon-v4-1-1-631/.

Ahlbrecht, P (2018) *… but we are all Facebook users!* 2018 [cited 4 May 2018]. Available from https://blog.onyxbits.de/but-we-are-all-facebook-users-715/.

Allen, J (2011) Powerful assemblages? *Area* vol. 43 (2):154-157. doi: 10.1111/j.1475-4762.2011.01005.x.

Aradau, C, and Huysmans, J (2014) Critical methods in International Relations: The politics of techniques, devices and acts. *European Journal of International Relations* vol. 20 (3):596-619. doi: 10.1177/1354066112474479.

Berry, D (2016) *The Philosophy of Software: Code and mediation in the digital age*. Basingstoke: Palgrave Macmillan.

Binns, R, et al. (2018) Third Party Tracking in the Mobile Ecosystem. In *WebSci '18 Proceedings of the 10th ACM Conference on Web Science*. Amsterdam, Netherlands: ACM Library.

Blanke, T (2014) *Digital Asset Ecosystems: Rethinking crowds and clouds*. Oxford: Chandos/Elsevier.

Brown, SD (2013) In praise of the parasite: the dark organizational theory of Michel Serres. *INFORMÁTICA NA EDUCAÇÃO: teoria & prática* 16 (1): 83-100, http://www.seer.ufrgs.br/index.php/InfEducTeoriaPratica/article/view/36928/25942).

Bucher, T (2016) Neither black nor box: ways of knowing algorithms. In *Innovative Methods in Media and Communication Research*, edited by Kubitschko, Sebastian and Kaun, Anne, 81-98. Basingstoke: Palgrave Macmillan.

Burns, R (2015) Rethinking big data in digital humanitarianism: Practices, epistemologies, and social relations. *GeoJournal* vol. 80 (4):477-490.

Burrell, J (2016) How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society* vol. 3 (1):2053951715622512.

Callon, M, and Latour, B (1981) Unscrewing the big Leviathan: how actors macro-structure reality and how sociologists help them to do so. In *Advances in Social Theory and Methodology: Toward an integration of micro-and macro-sociologies*, edited by Knorr-Cetina, Karin and Cicourel, A. V., 277-303. London: Routledge.

Coleman, EG (2001) High-Tech Guilds in the Era of Global Capital. *Anthropology of Work Review* vol. 22 (1):28-32.

Coleman, G (2011) Hacker politics and publics. *Public Culture* vol. 23 (3 65):511-516.

Crawford, K (2015) Can an Algorithm be Agonistic? Ten Scenes from Life in Calculated Publics. *Science, Technology & Human Values*. doi: 10.1177/0162243915589635.

Diakopoulos, N (2014) *Algorithmic Accountability Reporting: The investigation of Black Boxes*. Columbia University 2014 [cited 7 June 2017]. Available from https://doi.org/10.7916/D8ZK5TW2.

Galloway, AR, and Thacker, E (2007) *The Exploit: A Theory of Networks*. Minneapolis: University of Minnesota Press.

Hayles, NK (2017) *Unthought: The Power of the Cognitive Nonconscious*. Chicago: University of Chicago Press.

Helmond, A (2015) The platformization of the web: Making web data platform ready. *Social Media+ Society* vol. 1 (2):2056305115603080.

Introna, LD (2015) Algorithms, governance, and governmentality: On governing academic writing. *Science, Technology & Human Values*. doi: 10.1177/0162243915587360.

Jacobsen, KL (2015) *The Politics of Humanitarian Technology: Good intentions, unintended consequences and insecurity*. London: Routledge.

Jordan, T (2017) A genealogy of hacking. *Convergence* vol. 23 (5):528-544. doi: 10.1177/1354856516640710.

Kelty, CM (2018) Hacking the social? In *Inventing the social*, edited by Marres, Noortje, et al. Manchester: Mattering Press.

Kitchin, R (2017) Thinking critically about and researching algorithms. *Information, Communication & Society* vol. 20 (1):14-29.

Latour, B (1988) *Science in Action. How to follow scientists and engineers through society*. Cambridge, Mass.: Harvard University Press.

Law, J (2004) *After Method: Mess in Social Science Research*. London: Routledge.

Law, J, and Ruppert, E (2013) The social life of methods: Devices. *Journal of Cultural Economy* vol. 6 (3):229-240. doi: 10.1080/17530350.2013.812042.

Light, B, Burgess, J, and Duguay, S (2018) The walkthrough method: An approach to the study of apps. *New Media & Society* vol. 20 (3):881-900. doi: 10.1177/1461444816675438.

Lury, C (2018) Introduction. Activating the present of interdisciplinary methods. In *Routledge Handbook of Interdisciplinary Research Methods*, edited by Lury, Celia , et al., 1-25. Abingdon: Routledge.

MacKenzie, D (2005) Opening the black boxes of global finance. *Review of international political economy* vol. 12 (4):555-576.

Marres, N (2015) Why map issues? On controversy analysis as a digital method. *Science, Technology, & Human Values* vol. 40 (5):655-686.

Marres, N (2017) *Digital sociology: The reinvention of social research*: John Wiley & Sons.

OCHA (2017) *Centre for humanitarian data: connecting people and data to improve lives* (5 January 2017). OCHA 2017 [cited 5 December 2017]. Available from https://www.unocha.org/story/centre-humanitarian-data-connecting-people-and-data-improve-lives.

Pasquale, F (2015) *The Black Box Society*. Cambridge, MA: Harvard University Press.

Pasquinelli, M (2008) *Animal Spirits: A Bestiary of the Commons*. Amsterdam: NAi Publishers / Institute of Network Cultures.

Paßmann, J, and Boersma, A (2017) Unknowing Algorithms. On Transparency of Unopenable Black Boxes. In *The Datafied Society: Studying Culture through Data*, edited by Schäfer, Mirko Tobias and van Es, Karin. Amsterdam: Amsterdam University Press.

Petzel, E (2015) *AirMapView* (20 April 2015). Airbnb 2015 [cited 2 May 2019]. Available from https://medium.com/airbnb-engineering/airmapview-a-view-abstraction-for-maps-on-android-4b7175a760ac.

Plantin, J-C, Lagoze, C, Edwards, PN, and Sandvig, C (2018) Infrastructure studies meet platform studies in the age of Google and Facebook. *New Media & Society* vol. 20 (1):293-310. doi: 10.1177/1461444816661553.

Pybus, J, Coté, M, and Blanke, T (2015) Hacking the social life of Big Data. *Big Data & Society* vol. 2 (2). doi: 10.1177/2053951715616649.

Read, R, Taithe, B, and Mac Ginty, R (2016) Data hubris? Humanitarian information systems and the mirage of technology. *Third World Quarterly* vol. 37 (8):1314-1331.

Rieder, B, and Röhle, T (2012) Digital Methods: Five Challenges. In *Understanding Digital Humanities*, edited by Berry, David M., 67-84. London: Palgrave Macmillan UK.

Rogers, R (2013) *Digital methods*. Cambridge, Massachusetts: MIT press.

Seaver, N (2014) *On Reverse Engineering. Looking for the cultural work of engineers* (28 January 2017) 2014 [cited 6 June 2017]. Available from https://medium.com/anthropology-and-algorithms/on-reverse-engineering-d9f5bae87812.

Serres, M (1982) *The Parasite*. Baltimore: The Johns Hopkins University Press.

Serres, M (1995) *Genesis*. Translated by James, Genevieve and Nielson, James. Ann Arbor: University of Michigan Press.

Serres, M (2017) *Geometry. The Third Book of Foundations*. Translated by Burks, Randolph. London: Bloombury.

Srnicek, N (2017) *Platform Capitalism*. Cambrige: Polity.

Venturini, T (2012) Building on faults: How to represent controversies with digital methods. *Public Understanding of Science* vol. 21 (7):796-812. doi: 10.1177/0963662510387558.

Venturini, T, Bounegru, L, Gray, J, and Rogers, R (2018) A reality check (list) for digital methods. *New Media & Society*:1461444818769236.